IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Appellant	: Taher ELGAMAL et al.	Art Unit	: 2135
Serial No.	: 09/920,801	Examiner	: Klimach, Paula W.
Filed	: August 3, 2001	Conf. No.	: 8214
Title	: CRYPTOGRAPHIC POLICY FILTERS AND POLICY CONTROL METHOD AND APPARATUS		

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

SUPPLEMENTAL BRIEF ON APPEAL

(1) Real Party in Interest

AOL, LLC., the assignee of this application, is the real party in interest.

(2) Related Appeals and Interferences

A Notice of Appeal was filed on June 8, 2006, and a corresponding Appeal Brief was filed on October 10, 2006. Prosecution was thereafter reopened. This Supplemental Brief on Appeal is filed after filing a Notice of Appeal requesting reinstatement of the June 8, 2006 Appeal, since the newly applied art suffers the same deficiencies previously articulated in the October 10, 2006 Appeal Brief. There are no related Interferences.

(3) Status of Claims

Claims 31-34, 36-43 and 45-48 are now pending, of which claims 31 and 40 are independent. Claims 1-30, 35, and 44 are canceled. All claims have been rejected, and all claims have been appealed.

(4) Status of Amendments

All claim amendments have been entered; the presently pending claims appear in Appendix B.

(5) Summary of Claimed Subject Matter

Independent claim 31 is directed to a method for controlling functions of an application program. The method includes accessing a policy file that includes an attribute portion configured to store one or more policy attributes and a value portion having one or more attribute values.[1] Each attribute value corresponds to a policy attribute and indicates whether an application program may access the function represented by the policy attribute.[2] Each policy file includes a signature portion with at least one digital certificate.[3] The method includes determining whether the policy file is unaltered based on the signature portion of the policy file.[4] The method also includes retrieving at least one of the attributes, and, for each retrieved attribute, an attribute value corresponding to the attribute from the policy file.[5] The method further includes determining whether a function represented by a retrieved attribute is permitted to be accessed by the application program and permitting the application program to access the function conditioned upon a determination that the policy file is unaltered.[6]

Independent claim 40 is directed toward an apparatus for controlling functions of an application program. The apparatus is configured to, access a policy file that includes an attribute portion configured to store one or more policy attributes and a value portion having one or more attribute values,[7] each attribute value corresponding to a policy attribute and indicating whether an application program may use a function capable of being performed by the application program and represented by the policy attribute,[8] and a signature portion including at least one digital certificate.[9] The apparatus also is configured to determine whether the policy file is unaltered based on the signature portion of the policy file[10] and retrieve at least one of the attributes and, for each retrieved attribute, an attribute value corresponding to the attribute from the policy file.[11] The apparatus is further configured for determining whether a function represented by a retrieved attribute is permitted to be accessed by the application program, and

---

[1] *See* e.g., specification, page 10, lines 1-18
[2] *See* e.g., specification, page 10, table 1
[3] *See* e.g., specification, page 12, lines 5-8
[4] *See* e.g., specification, page 12, lines 8-12
[5] *See* e.g., specification, page 11, table 2
[6] *See* e.g., specification, page 14, lines 22-34
[7] *See* e.g., specification, page 10, lines 1-18
[8] *See* e.g., specification, page 10, table 1
[9] *See* e.g., specification, page 12, lines 5-8
[10] *See* e.g., specification, page 12, lines 8-12
[11] *See* e.g., specification, page 11, table 2

for permitting the application program to access the function conditioned upon a determination that the policy file is unaltered. [12]

(6) Grounds of Rejection

a. Claims 31, 36-40, and 45-48 under 35 U.S.C. § 103

Claims 31, 36-40, and 45-48 under 35 U.S.C. § 103 as being unpatentable over Drews (U.S. Patent No. 6,647,494) in view of Challener (U.S. Patent No. 7,096,496) and further in view of Perona (U.S. Patent No. 6,671,809).

b. Claims 32-34 and 41-43 under 35 U.S.C. § 103

Claims 32-34 and 41-43 also were rejected under 35 U.S.C. § 103 as being unpatentable over Drews in view of Perona, and further in view of Anderl (WO 87/07063).

(7) Argument

a. Claims 31, 36-40, and 45-48 under 35 U.S.C. § 103 are not properly rejected under 35 U.S.C. § 103 as being unpatentable over Drews in view of Challener and further in view of Perona.

Following the filing of the previous Appeal Brief,[13] prosecution was reopened and a new reference (Challener) was added to the previously applied combination of references. As the new reference does not overcome the deficiencies identified in the previous Appeal Brief, this Supplemental Appeal Brief is being filed to further address the new reference.

Discussion of the claimed subject matter's utility and the new reference's deficiencies is included below. Thereafter, Appellant's previous discussion regarding the previously applied references is included and updated for convenience.

In a variety of contexts, it may be advisable to distribute multiple versions of packaged software, making selected functions available or unavailable based upon factors involving the end-user or recipient to which the packaged software is distributed (e.g., the end-user's country status as domestic or foreign to the software company's home country). Particular utility is demonstrated when considering that various countries impose differing limits on encryption

[12] *See* e.g., specification, page 14, lines 22-34
[13] A Notice of Appeal was filed on June 8, 2006, and a corresponding Appeal Brief was filed on October 10, 2006

capabilities of software, or the exportation thereof. In order to comply with the various regulations, a software company may develop, control, and selectively distribute many different versions of a single type of software, each of which including different capabilities to address restrictions imposed by the particular country to which it is destined. For example, to comply with such restrictions, a company may develop and separately service/distribute each of several different versions of their software, including software capable of 128 bit encryption to be delivered to a country mandating that encryption not be stronger than 128 bit, and software capable of 256 bit encryption to be delivered to countries without such regulations.

As an alternative to such selective distribution, the company could implement rules that check various factors to determine whether a function is to be made available to an end user. For example, functions may selectively be made available based on the identity of an end-user or the geography of their use, and the rules may allow or disallow otherwise available functions based on the detected geography or identity (e.g., disallow 256 bit encryption when operating in a restricted or restrictive country). However, such rules may be easily circumvented by individuals who trick their machines into erroneous determinations of user identity or geography of use (e.g., that the machine is in the U.S. rather than a different country).

The presently pending claims enable distribution of software in a manner that enables control by the distributor of the software's function during use while relieving the distributor of multiple different software versions or risks of user circumvention of controls. Specifically, the presently pending claims refer to the use of attribute values, each of which corresponding to a policy attribute and indicating whether an application program may use a function capable of being performed by the application program and represented by the policy attribute.

In keeping with the use case illustrated above, one implementation of this approach allows software to be distributed to a regulated country with an attribute value set to indicate availability of 64 bit encryption but not 256 bit encryption (or the end-user would provide an attribute value with a hardware plug-in, such as a Fortezza card or other cryptographic smart card). Depending on the attribute value, the end-user or system may perceive a version that appears able only to handle 64 bit encryption, notwithstanding its actual capability of handling 256 bit encryption, but for the attribute.

Although the rules and attribute values may also be used (together or separately) with the attribute value approach presently claimed to achieve similar goals, the approaches clearly achieve the goals in different ways and have differing capabilities. One manner in which the attribute approach is differentiated from the rules-based approach is that, through the attribute approach, recipients would not be able to circumvent the desired restriction by merely changing the apparent location of their computer operation, such as is available for circumvention of the rules-based approach. The attribute based approach is further differentiated from the rules-based approach inasmuch as attributes values may be made difficult to alter, such as when using a hardware encryption card.

Independent Claim 31 and Dependent Claims 36-39

Appellant requests reversal of the rejection to claim 31, as a consequence of Drews, Challener, Perona, and the proposed combination of these references failing to teach or suggest the use of attribute values, each of which corresponding to a policy attribute and indicating whether an application program may use a function capable of being performed by the application program and represented by the policy attribute, as recited by claim 31.

As revealed through the following excerpt, the Final Office Action relies on Challener to describe claim language argued in the previous Appeal Brief:

> **Challener discloses a policy file wherein each attribute value corresponds to a policy attribute and indicat[es] whether an application program may use a function (Fig. 4)...[I]t would have been obvious...to use a profile that indicates the level of access to the user as in Challener [See page 4]**

By this excerpt, it appears as though Challener is being relied upon to disclose or somehow otherwise render obvious the claim limitation concerning attributes, namely the requirement for "each attribute value corresponding to a policy attribute and indicating whether an application program may use a function capable of being performed by the application program and represented by the policy attribute." Appellants disagree.

Challener discloses operating system security software which regulates access of users through rules instead of the claimed attribute values.[14] Specifically, in Challener, regulating

[14] *See* Challener, column 2, lines 13-22

authorization of users includes use of rule-based determinations in view of security profiles.[15] A security profile includes various fields, such as, for example, tamper evidence, unsuccessful attempts (unsuccessful login attempts), and a level of access.[16] The security profile is stored on the personal computer, and can be adjusted to increase or decrease security for a user.[17]

While Challener uses rules and security profiles to provide a flexible system that may be adapted to particular users, such flexibility does not allow for distribution of software with certain included functions enabled and others disabled. Rather, by design, the Challener rules deal exclusively with enabled functions of software, dictating their availability on a user-by-user basis depending upon application of the Challener rules against each user's security profile. In particular, Challener's level of access indicates "a level of authorization for the individual or a level of a security risk" through more or less secure states.[18]

Thus, Challener deals exclusively with enabled software functions, providing rules that merely decide which of the enabled functions are to be made available to any particular user based on the security profile, e.g., of that user. By contrast, the claimed attribute values indicate whether an application (as opposed to a particular user) "may use" a function that is capable of being performed – i.e., they indicate whether the application function is enabled and this preempts the decision – making logic of Challener. In greater detail, pending claim 31 recites attribute values that are said to indicate whether an application program may use a function that the program is otherwise capable of performing. In particular, the claim requires that each attribute value indicates whether an application may use a function represented by the policy attribute.

Referring back to the cryptographic example above, this distinction of the claim language allows specific control of functions through use of multiple attribute values. In particular, the claim language of "each attribute value...indicating whether an application program may use a function...represented by the policy attribute" enables the determination of whether each cryptographic function (or other function) is permitted for use by any and all users of the software, without regard for rules that account for a general or specific level of confidence in a

[15] *See* Challener, column 6, lines 24-29 and FIG. 4
[16] *See* Challener, FIG. 4, elements 72, 74, and 76
[17] *See* Challener, column 6, lines 29-37
[18] *Id.*

user. Further, this determination may be made (through use of attribute values) prior to the loading of the software on the device, and may prevent some tampering or tricking of the software. For example, in one representative embodiment mentioned for illustrative purposes only, the attribute values are stored on a separate cryptographic piece of hardware that may be plugged into the hardware running the application (e.g., a Fortezza card). In such an embodiment, the attribute values are read off the cryptographic piece of hardware to perform the claimed method. In this manner, a software vendor may distribute substantially identical versions of software, while controlling sensitive functions through separate distribution of cryptographic hardware to be plugged into the hardware running the application.

Moreover, and further evidencing this point of distinction, the presently-claimed attribute values may be used together with and complementary to the alternative user-based security policy and rules of Challener to enable user-independent disabling of application program function (e.g., by way of the claimed attribute values) and user-specific control over application program functions (e.g., by way of the Challener user-based security policies and rules). Consequently, Challener's level of access represents a rule-based user-dependent approach that fails to teach or suggest the claimed attribute values, which each correspond to a policy attribute and indicate whether an application program may use a function capable of being performed by the application program and represented by the policy attribute.

As an additional point of distinction which exemplifies the differences between Challener's rule-based approach and the claimed attribute values, Challener's level of access does not represent a function, nor does it correspond to an attribute that represents a function. Specifically, the claim requires that each attribute value correspond to a policy attribute representing a function. ("each attribute value corresponding to a policy attribute and indicating whether an application program may use a function capable of being performed by the application program and represented by the policy attribute"). As described above, the claimed representation of a function is important in that it enables the attribute values to have the claimed effect upon the functions. Rather than representing a function, Challener's level of access "either indicates a level of authorization for the individual or a level of a security risk"[19] as part of

[19] *See* Challener, column 6, lines 26-29

Challener's rule-based approach. More specifically and further to the point that Challener's level of access represents something other than a function, Challener details that the level of access represents "a more secure state" with a binary value of 1, and "a less secure state" with a binary value of 0.[20] Neither a level of authorization nor a level of a security risk indicates whether an application program may use a function capable of being performed by the application program and represented by the policy attribute, as required by the claim language.

Examiner Comments During the March 13, 2007 Interview

Appellant notes that during the interview, the Examiner suggested that if not currently proper, the applied rejection would be proper if the same combination of references were applied differently. Specifically, the Examiner suggested that an argued deficiency of one reference (in view of the applied rejection) could be corrected by rewriting the rejection so the deficiency is rejected in view of another of the applied references. Appellant notes that as none of the references, alone or in combination, teach or suggest "each attribute value corresponding to a policy attribute and indicating whether an application program may use a function capable of being performed by the application program and represented by the policy attribute," no possible combination of the applied references could be made into a valid rejection.

Independent Claim 40 Dependent Claims 45-48

Similar to claim 31, independent claim 40 recites a policy file that includes attribute values, each attribute value corresponding to a policy attribute and indicating whether an application program may use a function capable of being performed by the application program and represented by the policy attribute. For the reasons above with respect to claim 31, appellants submit that the rejection of independent claim 40 and dependent claim 45-48 should be withdrawn.

b. Claims 32-34 and 41-43 are not properly rejected under 35 U.S.C. § 103(a) as being unpatentable Drews in view of Perona, further in view of Anderl.

As claims 32-34 and 41-43 are all dependent on independent claim 31 or independent claim 40, appellant requests reversal of this rejection because neither Drews, Perona, Anderl, nor any proper combination of the references teaches or suggests the subject matter of the

[20] *See* Challener, column 6, lines 38-41

independent claims 31 and 40. For example, neither Drews, Perona, nor Anderl teach or suggest "each attribute value corresponding to a policy attribute and indicating whether an application program may use a function capable of being performed by the application program and represented by the policy attribute and represented by the policy attribute." Specifically, Anderl does not make up for the deficiencies of Drews in view of Perona, nor does the Final Office Action contend so.

For at least these reasons, appellant requests reversal of the § 103 rejection of claims 32-34 and 41-43.

c. Deficiencies of the references discussed in the previous Appeal Brief resulting in reopening of prosecution.

Drews does not teach or suggest the claimed attribute values

Drews discloses a method and system for checking authorization for program additions or updates. In order to ensure authorization of the delivered program, a client sends authorization information to the host, which uses the authorization information to generate and deliver the program additions or updates to the client.[21] Drews refers to the authorization information as an "update token," and the generated and delivered program addition or update as a "credential manifest."[22]

Among other aspects of Drew, the Office Action references its disclosure of the "manifests." A manifest (i.e. "request credential manifest") includes an update token, a list of configurable parameters to be updated, a list of new values for those configurable parameters, and a manifest digital signature. The "update token is a hash value that the console platform needs [in order] to construct a request credential manifest." As for the other components of Drew's manifest, the list of configurable parameters is the software to be updated or added and the manifest digital signature is used for verification. None are attributes that indicate whether an application program may use a function.

Stated differently, the claimed policy file requires attribute values which indicate "whether an application program may use a function...represented by the policy file," and the

[21] *See* Drews, FIG. 3
[22] *See* Drews, column 3, line 3 to column 4, line 17

manifest of Drews simply does not include such attribute values, nor does the Office Action contend so. In fact, to the contrary, the Office Action acknowledges the failure of Drews to include attributes revealing whether an application can use a feature, stating:

> **Drews does not disclose expressly disclose [sic] determining whether an application program may use a function capable of being performed by the application program and thus determining whether a function represented by a received attribute is permitted to be accessed by the application program; and permitting the application program to access the function conditioned upon the determination that the policy file is unaltered. [See pages 3-4]**

Appellants acknowledge these shortcomings of Drews with particular reference to the claim limitation of "each attribute value corresponding to a policy attribute and indicating whether an application program may use a function capable of being performed by the application program and represented by the policy attribute."

Perona does not teach or suggest the claimed attribute values

Like Drews and Challener, Perona fails to teach or suggest the claim limitation concerning attributes, namely the requirement for "each attribute value corresponding to a policy attribute and indicating whether an application program may use a function capable of being performed by the application program and represented by the policy attribute." The Office Action does not suggest otherwise, instead relying upon Perona to reject the remainder of the claim language:

> **Perona discloses determining whether an application program may use a function capable of being performed by the application program and thus determining whether a function represented by a retrieved attribute is permitted to be accessed by the application program (column 6 lines 15-23). [See page 4]**

Specifically, Perona generally relates to open architecture software that enables software to be loaded on a platform, such as a cellular phone, by performing checks among system components based on configuration and rule information.[23] In Perona, software modules may be installed to update or add to an application on a platform. The software modules include rules with requirements that "must be met by the modules" for the platform to be able to load the modules.[24] In particular, the software modules include pointer record rules specifying what are

[23] *See* Perona, column 1, lines 7-13 and FIG. 1
[24] *See* Perona, column 4, lines 14-16

the specific requirements and limitations imposed on a given module.[25] Notably, in Perona, the rules are read, and based on the rules, a set of requirements or limitations are assigned, as necessary, for installation to take place. If the device is able to meet the requirements or limitations assigned by the rules, the platform loads the module for execution.[26]

While Perona's rules enable a process to determine what requirements or limitations are needed to be met in order to determine whether or not a target computer can load or use an application, nowhere does Perona teach or suggest using attributes values which specifically indicate (or used in a look-up table to indicate) whether an application program may use a function capable of being performed by the application program and represented by the policy attribute.

More specifically, the cited portion of Perona describes rules that are used to construct a process to assign module requirements or constraints that must be met before the module can be loaded onto the platform. Notably, while the claim limitation requires the use of attribute values indicating whether an application program may use a function, Perona uses rules to create a process to determine whether application specific requirements are met. Specifically, the rules of Perona enable the device to develop a process to determine whether or not factors *external to the rules* permit the application to be loaded. For example, Perona details a wireless information transmitting system where the rules (i.e., module requirement) specify a waveform requirement of an RF module. "Before the platform 20 can load the referenced module, the platform must verify the limitations and requirements specified in the module rules record 46 are met."[27] In contrast, the claimed attribute values themselves "[indicate] whether an application program may use a function."

This distinction is not trivial. Rather, it is quite important, as it allows the claimed invention to achieve results not attainable by Perona. By way of example, the specification details an implementation in which software is loaded with multiple available functions, and with the determination of which functions are permitted stored in the attribute values.[28] In this

---

[25] *See* Perona, column 4, lines 26-32
[26] *See* Perona, column 4, lines 21-43
[27] *See* Perona, column 4, lines 37-43
[28] *See e.g.*, application, FIG. 2

manner, the determination of whether the functions are permitted is made prior to the loading of the software on the device, may prevent some tampering or tricking of the software.

Therefore, claim 31 defines an invention that is patentable over Drews in view of Challener and further in view of Perona, as do pending dependent claims 36-39. Accordingly, appellants requests reconsideration and withdrawal of the imposed rejection.

As a more specific example of the differences between Challener's indication of a level of access or security risk and Perona's use of rules and module requirements as compared to the claimed attribute values, dependent claim 38 describes one implementation in which a truth expression is included in each of the attribute values where the truth expression is one of a true flag, a false flag, and a conditional flag which may enable, disable, or conditionally enable an available function of a program.

d. Conclusion.

For the foregoing reasons, the rejections should be reversed.

In accordance with appellant's Notice of Appeal filed April 04, 2007, appellant submits this Supplemental Appeal Brief.

Please apply any other charges or credits to Deposit Account No. 06-1050.

Respectfully submitted,

Date: August 31, 2007

/Gabriel Olander/
Gabriel D. Olander
Reg. No. 59,185

Fish & Richardson P.C.
1425 K Street, N.W.
11th Floor
Washington, DC 20005-3500
Telephone: (202) 783-5070
Facsimile: (202) 783-2331

40403147.doc

Appendix of Claims

1-30. (Cancelled)

31. A method for controlling functions of an application program, the method comprising:

accessing a policy file that includes an attribute portion configured to store one or more policy attributes and a value portion having one or more attribute values, each attribute value corresponding to a policy attribute and indicating whether an application program may use a function capable of being performed by the application program and represented by the policy attribute, and a signature portion including at least one digital certificate;

determining whether the policy file is unaltered based on the signature portion of the policy file;

retrieving at least one of the attributes and, for each retrieved attribute, an attribute value corresponding to the attribute from the policy file;

determining whether a function represented by a retrieved attribute is permitted to be accessed by the application program;

permitting the application program to access the function conditioned upon a determination that the policy file is unaltered.

32. The method of claim 31 wherein the policy file comprises a JAVA archive file.

33. The method of claim 31 wherein the policy file comprises multiple component files, at least one of the component files storing some of the attribute portions and attribute values.

34. The method of claim 33 wherein at least one of the multiple component files is associated with a signature portion including at least one digital certificate for ensuring that the policy file has not been modified and a signature portion including at least one digital certificate for ensuring that the policy file has not been modified and applying to a particular component file.

35. (Cancelled)

36. The method of claim 31 wherein:

the signature portion applies to the attribution portion and the value portion of the policy file;

determining whether the policy file is unaltered comprises determining whether the attribute portion and the value portion are unaltered based on the signature portion.

37. The method of claim 36 wherein the signature portion applies to the policy file.

38. The method of claim 31 wherein:

each of the attribute values is one of a string, an integer number, and a truth expression.

39. The method of claim 38 wherein the truth expression is one of a true flag, a false flag, and a conditional flag.

40. An apparatus for controlling functions of an application program, the apparatus being configured to:

access a policy file that includes an attribute portion configured to store one or more policy attributes and a value portion having one or more attribute values, each attribute value corresponding to a policy attribute and indicating whether an application program may use a function capable of being performed by the application program and represented by the policy attribute, and a signature portion including at least one digital certificate;

determine whether the policy file is unaltered based on the signature portion of the policy file;

retrieve at least one of the attributes and, for each retrieved attribute, an attribute value corresponding to the attribute from the policy file;

determining whether a function represented by a retrieved attribute is permitted to be accessed by the application program; and

permitting the application program to access the function conditioned upon a determination that the policy file is unaltered.

41. The apparatus of claim 40 wherein the policy file comprises a JAVA archive file.

42. The apparatus of claim 40 wherein the policy file comprises multiple component files, at least one of the component files storing some of the attribute portions and attribute values.

43. The apparatus of claim 42 wherein at least one of the multiple component files is associated with a signature portion including at least one digital certificate for ensuring that the policy file has not been modified and the signature portion applying to a particular component file.

44. (Cancelled)

45. The apparatus of claim 40 wherein the signature portion applies to the attribute portion and the value portion of the policy file;

determining whether the policy file is unaltered comprises determining whether the attribute portion and the value portion are unaltered based on the signature portion.

46. The apparatus of claim 45 wherein the signature portion applies to the policy file.

47. The apparatus of claim 40 wherein:

each of the attribute values is one of a string, an integer number, and a truth expression.

48. The apparatus of claim 47 wherein the truth expression is one of a true flag, a false flag, and a conditional flag.

Evidence Appendix

None

Related Proceedings Appendix

None